

## **LISTING OF CLAIMS**

This listing of claims replaces all prior listings of claims in this application.

1. (Previously Presented) A method for emulating the execution of a target program comprising instructions of an instruction set of a target computer on a host computer having a different instruction set, said method comprising:  
performing a static translation of the instructions of the target program into a series of instructions of an intermediate instruction set, the intermediate instruction set being optimized for interpretation on the host computer; and  
executing the series of instructions of the intermediate instruction set by interpretation directly by the host computer.
2. (Original) The method recited in claim 1, wherein the intermediate instruction set comprises a plurality of control words that are derived, at least in part, by mapping control words of the instruction set of the target machine into the fundamental word size of the host machine.
3. (Original) The method recited in claim 1, wherein the intermediate instruction set comprises a plurality of control words that are derived by redefining control words of the target computer to minimize the number of masking and shifting operations needed to decode the plurality of control words of the intermediate instruction set.
4. (Original) The method recited in claim 1, wherein the intermediate instruction set comprises a plurality of different types of control words having formats defined to minimize the time needed to determine the type of a control word.
5. (Original) The method recited in claim 1, wherein the intermediate instruction set comprises a plurality of controls words derived from control words of the instruction set of the target machine in a manner that reduces the number of different forms of control words in the intermediate instruction set.

6. (Original) The method recited in claim 1, wherein a code structure of the intermediate instruction set comprises code words have a fixed length that matches the fundamental word size of the host machine.
7. (Original) The method recited in claim 1, wherein the instructions of the intermediate instruction set have a fixed length and do not cross code word boundaries.
8. (Original) The method recited in claim 1, wherein zero-address instructions of the instruction set of the target machine for pushing data onto a stack for use in a subsequent zero-address instruction operation are incorporated as explicit addresses into a new instruction in the intermediate instruction set for performing that operation, thereby reducing the number of different instructions in the intermediate instruction set.
9. (Previously Presented) An emulation system for emulating the execution of a target program comprising instructions of an instruction set of a target computer on a host computer having a different instruction set, said emulator comprising;
  - a code translator that performs a static translation of the instructions of the target program into a series of instructions of an intermediate instruction set, the intermediate instruction set being optimized for interpretation on the host computer; and
  - an interpreter that executes the series of instructions of the intermediate instruction set by interpretation directly on the host computer.
10. (Original) The emulation system recited in claim 9, wherein the intermediate instruction set comprises a plurality of control words that are derived, at least in part, by mapping control words of the instruction set of the target machine into the fundamental word size of the host machine.
11. (Original) The emulation system recited in claim 9, wherein the intermediate instruction set comprises a plurality of control words that are derived by redefining control words of the target computer to minimize the number of masking and shifting operations needed to decode the plurality of code words of the intermediate instruction set.

12. (Original) The emulation system recited in claim 9, wherein the intermediate instruction set comprises a plurality of different types of control words having formats defined to minimize the time needed to determine the type of a control word.
13. (Original) The emulation system recited in claim 9, wherein the intermediate instruction set comprises a plurality of control words derived from control words of the instruction set of the target machine in a manner that reduces the number of different forms of control words in the intermediate instruction set.
14. (Original) The emulation system recited in claim 9, wherein a code structure of the intermediate instruction set comprises code words having a fixed length that matches the fundamental word size of the host machine.
15. (Original) The emulation system recited in claim 9, wherein the instructions of the intermediate instruction set have a fixed length and do not cross code word boundaries
16. (Original) The emulation system recited in claim 9, wherein the code translator runs as a user mode process under control of a host operating system on the host computer, and wherein the interpreter runs as a kernel mode driver thread under the host operating system.
17. (Original) The emulation system recited in claim 9, wherein the emulation system may comprise multiple instances of the interpreter each running as a different thread in the kernel space of the host operating system.
18. (Original) The emulation system recited in claim 9, wherein zero-address instructions of the instruction set of the target machine for pushing data onto a stack for use in a subsequent zero-address instruction operation are incorporated as explicit addresses into a new instruction in the intermediate instruction set for performing that operation, thereby reducing the number of different instructions in the intermediate instruction set.

19. (Previously Presented) A computer-readable medium having stored thereon program code that when executed by a host computer enables the host computer to emulate the execution of a target program comprising instructions of an instruction set of a target computer on the host computer, wherein the host computer has a different instruction set, by:
- performing a static translation of the instructions of the target program into a series of instructions of an intermediate instruction set, the intermediate instruction set being optimized for interpretation on the host computer; and
  - executing the series of instructions of the intermediate instruction set directly by interpretation on the host computer.
20. (Original) The computer-readable medium recited in claim 19, wherein the intermediate instruction set comprises a plurality of control words that are derived, at least in part, by mapping control words of the instruction set of the target machine into the fundamental word size of the host machine.
21. (Original) The computer-readable medium recited in claim 19, wherein the intermediate instruction set comprises a plurality of control words that are derived by redefining control words of the target computer to minimize the number of masking and shifting operations needed to decode the plurality of control words of the intermediate instruction set.
22. (Original) The computer-readable medium recited in claim 19, wherein the intermediate instruction set comprises a plurality of different types of control words having formats defined to minimize the time needed to determine the type of a control word.
23. (Original) The computer-readable medium recited in claim 19, wherein the intermediate instruction set comprises a plurality of controls words derived from control words of the instruction set of the target machine in a manner that reduces the number of different forms of control words in the intermediate instruction set.

24. (Original) The computer-readable medium recited in claim 19, wherein a code structure of the intermediate instruction set comprises code words have a fixed length that matches the fundamental word size of the host machine.

25. (Original) The computer-readable medium recited in claim 19, wherein the instructions of the intermediate instruction set have a fixed length and do not cross code word boundaries.

26. (Original) The computer-readable medium recited in claim 19, wherein zero-address instructions of the instruction set of the target machine for pushing data onto a stack for use in a subsequent zero-address instruction operation are incorporated as explicit addresses into a new instruction in the intermediate instruction set for performing that operation, thereby reducing the number of different instructions in the intermediate instruction set.

27. (Previously Presented) A method for defining an intermediate instruction set based on the instruction set of a target machine for use in an emulation system in which a target program, which comprises instructions of the target machine instruction set, is executed by emulation on a host computer having a different instruction set by (i) performing a static translation of the instructions of the target program into series of instructions of the intermediate instruction set, and then (ii) executing the series of instruction set directly by interpretation on the host computer, wherein the intermediate instruction set is optimized for interpretation on the host computer, said method comprising:

mapping control words of the instruction set of the target machine into the fundamental word size of the hose machine to derive a set of control words of the intermediate instruction set;

redefining control words of the instruction set of the target machine to reduce the number of different forms of control words in the intermediate instruction set; and

defining a code structure of the intermediate instruction set in which code words of that structure have a fixed length that matches the fundamental word size of the host machine.

28. (Original) The method recited in claim 27 wherein said step of mapping control words further comprises redefining control words of the instruction set of the target computer in the intermediate instruction set to minimize the number of masking and shifting operations needed to decode the control words of the intermediate instruction set.

29. (Original) The method recited in claim 27 wherein said step of mapping control words further comprises redefining control words of the instruction set of the target computer in the intermediate instruction set to minimize the time needed to determine the type of a control word of the intermediate instruction set.

30. (Original) The method recited in claim 27, further comprising defining a set of instructions of the intermediate instruction wherein the instructions have a same fixed length and do not cross code word boundaries.

31. (Original) The method recited in claim 27, further comprising incorporating zero-address instructions of the target machine instruction set for pushing data onto a stack for use in a subsequent zero-address instruction operation, as explicit addresses in a new instruction in the intermediate instruction set for performing that operation, thereby reducing the number of different instructions in the intermediate instruction set.

32. (Currently Amended) An emulation system for emulating the execution of a target program comprising instruction of an instruction set of a target computer on a host computer having a different instruction set, said ~~emulator~~ emulation system comprising: ~~and~~  
a code translator that performs a static translation of the instructions of the target program into a series of instructions of an intermediate instruction set, the intermediate instruction set being optimized for interpretation on the host computer; and  
an interpreter that executes the series of instructions of the intermediate instruction set by interpretation on the host computer, ~~and~~  
wherein the code translator runs as a user mode process under control of a host operating system on the host computer, and wherein the interpreter runs as a kernel mode driver thread under the host operating system.

33. (Currently Amended) An emulation system for emulating the execution of a target program comprising instruction of an instruction set of a target computer on a host computer having a different instruction set, said ~~emulator~~ emulation system comprising:

a code translator that performs a static translation of the instructions of the target program into a series of ~~instruction~~ instructions of an intermediate instruction set, the intermediate instruction set being optimized for interpretation on the host computer; and

an interpreter that executes the series of instructions of the intermediate instruction set by interpretation on the host computer,;

wherein the emulation system may comprise multiple instances of the interpreter each running as a different thread in the kernel space of the host operating system.